Comparing and Evaluating HMM Ensemble Training Algorithms Using Train and Test and Condition Number Criteria.

Richard I. A. Davis and Brian C. Lovell

The Intelligent Real-Time Imaging and Sensing (IRIS) Group,

School of Information Technology and Electrical Engineering,

The University of Queensland,

Brisbane, Australia 4072

Abstract

Hidden Markov Models have many applications in signal processing and pattern recognition, but their convergence-based training algorithms are known to suffer from over-sensitivity to the initial random model choice. This paper describes the boundary between regions in which ensemble learning is superior to Rabiner's multiple-sequence Baum-Welch training method, and proposes techniques for determining the best method in any arbitrary situation. It also studies the suitability of the training methods using the condition number, a recently proposed diagnostic tool for testing the quality of the model. A new method for training Hidden Markov Models

---

Correspondence to: <**riadavis@itee.uq.edu.au**>

called the Viterbi Path Counting algorithm is introduced and is found to produce significantly better performance than current methods in a range of trials.

Key Words: Ensemble learning, Baum-Welch Algorithm, hidden markov model, multiple sequence, model structure, Viterbi

---

1 Introduction

Hidden Markov Models have applications in many areas from speech recognition [1, 2, 3, 4], to computer vision including gesture recognition [5], face recognition [6], and handwriting recognition [7].

Previous studies [8, 9, 10] by the authors have shown that ensemble learning is superior to Rabiner's multiple-sequence learning method ([1], Chapter 2, p 273) when applied to a 5-state model involving short observation sequences and left-right models, which are of particular interest in speech recognition applications. However, it was noticed in some trials that when training on a single sequence, Rabiner's method was superior to ensemble learning, and no benefits were to be found by forming an ensemble of models, each initialized randomly and trained on a copy of that single sequence.

This suggested that there may be a boundary in parameter space separating regions in which Rabiner's method was superior to ensemble learning. The purpose of this paper is to investigate the general shape and location of this boundary, and to suggest a procedure for determining which method to use in a given situation.

Rcently McCane and Caelli [11] introduced techniques for understanding the general

quality of the structure of any HMM based upon the rank of the combined transition and emission matrices. This paper also includes a comparison of this methodology with the well-established technique of train-and-test for determining the effectiveness of different training algorithms.

A theme of this paper is that there are essentially three different ways of training an HMM from multiple training sequences irrespective of the update equations. These are as follows:

- Each iterative update takes into account all of the training data in a single step (e.g., Rabiner and Juang's multi-sequence training method [2], implemented in the Bayes Net Toolbox [21]).

- Training is done separately on separate models for each sequence, and then the models are combined (e.g., Mackay's [12] ensemble methods as examined in [8], [9], [10]).

- Separate training sequences are alternately used by the training algorithm, and a common underlying model is updated with each presented sequence (e.g., Viterbi Path Counting algorithm as described in this paper).

In order to study the mechanism of HMM training, this paper focuses upon established training mechanisms (Baum-Welch and Ensemble Averaging) as well as an algorithm called Viterbi Path Counting (which is related to Stolcke and Omohundro's Best-First model merging algorithm[14] which merges models using Viterbi's algorithm to form sequence counts) to represent these three distinct methods. These are investigated in terms of their relative performance in different regions of the parameter space for the model size,

model structure, the length of the sequences, and the number of distinct sequences.

The paper concludes with a study showing the relationship between the inverse condition number of the initial generating model and the inverse condition number of the models produced by each of the three methods: Baum-Welch, Ensemble Averaging, and Viterbi Path Counting. The results of a gesture recognition trial using these three methods are also mentioned briefly.

2 Background and Notation

A hidden Markov model ([2], Chapter 6) consists of a set of N nodes, each of which is associated with a set of M possible observations. The parameters of the model include 1) an initial state

$$\pi = [p_1, p_2, p_3, ..., p_N]^T$$

with elements $p_n, \; n \in [1, N]$ which describes the distribution over the initial node set, 2) a transition matrix

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1N} \\ a_{21} & a_{22} & \ldots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \ldots & a_{NN} \end{pmatrix}$$

with elements a_{ij} with $i, j \in [1, N]$ for the transition probability from node i to node j given that the HMM is currently in state i, where $i \in [1, N]$ and 3) an observation matrix

$$
\mathbf{B} = \begin{pmatrix} b_{11} & b_{12} & \ldots & b_{1M} \\ b_{21} & b_{22} & \ldots & b_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ b_{N1} & b_{N2} & \ldots & b_{NM} \end{pmatrix}
$$

with elements b_{im} denoting the probability of observing symbol $m \in [1, M]$ given that the system is in state $i \in [1, N]$. Rabiner and Juang denote the HMM model parameter set by $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$.

The model order pair (N, M) together with additional restrictions on allowed transitions and emissions defines the structure of the model (see figure 1 for an illustration of two different transition structures).

The Baum-Welch algorithm is an "iterative update" algorithm which re-estimates parameters of a given hidden Markov model to produce a new model which has a higher probability of generating the given observation sequence. This re-estimation procedure is continued until no more significant improvement in probability can be obtained and the local maximum is thus found. Alternatively, in some applications a fixed number of iterations is used to prevent over-fitting to the training data. This paper focuses upon the stopping criteria rather than a fixed number of iterations. The re-estimation procedure is initiated with a random set of HMM parameters that match the known structural constraints on the HMM (using Left-Right models only). This guarantees that every step of the convergence generates a left-right model. However similar results are often obtained

when no transition constraints are placed upon the initial model.

For example, if it is known that the model is left-right and always starts in state 1, set $\pi = [1, 0, 0, \ldots, 0]^T$ and A to upper-triangular. Other elements of A and B are then set to random values with the constraint that the row sums must equal unity to ensure that rows can be treated as probability mass functions. Note that elements of the matrices A and B that are set to zero will remain zero after re-estimation due to the nature of the algorithm. Hence the structure is preserved throughout the procedure. If the starting node is not known, then initially the elements of π can be set randomly with the constraint that the sum must equal unity as before. After the re-estimation runs to completion, one element, π_j say, will be close to unity and all other elements will be negligible. The index j would then represent an estimate of the starting node for this sequence.

2.1 Rabiner and Juang Multiple Sequence Method

Now consider the case where K observation sequences are known to be generated by the same HMM and the objective is to determine the HMM parameters that yield high probability of generating all K observed sequences.

Rabiner and Juang [2] proposed just such a multi-sequence training algorithm using the K observation sequences at each stage of the Baum-Welch re-estimation to iteratively update a single HMM parameter set. The re-estimation formula for this type of iterative update method is as follows (reproduced from [2]):

$$\overline{a}_{ij} = \frac{\sum_k W_k \sum_{t=1}^{T_k} \alpha_i^k a_{ij} b_j(O_{t+1}^{(k)}) \beta_{t+1}^{(k)}(j)}{\sum_k W_k \sum_{t=1}^{T_k} a_t^k(i) \beta_t^k(i)} \tag{1}$$

$$\overline{b}_{ij} = \frac{\sum_k W_k \sum_{O_t(k)=v_j} \alpha_t^k(i) \beta_t^k(i)}{\sum_k W_k \sum_{t=1}^{T_k} \alpha_t^k(i) \beta_t^k(i)} \tag{2}$$

where $W_k = 1/P_k, k \in [1 \ldots k]$ is the inverse of the probability of the current model estimate generating training sequence k, evaluated using the forward algorithm [2]. The Forward and Backward algorithms are used to calculate $\alpha_t^k(i)$ and $\beta_t^k(i)$ using the observation substrings $O(1 \ldots t)$ and $O(t+1 \ldots T)$ respectively.

The form of the re-estimation relation for π depends upon the model structure and is trivial in the case of left-right models (the only case covered in [2]), where $\pi = [1, 0, 0, \ldots, 0]^T$ by assumption. For other models, such as cyclic (see figure 1), one method is to run the Baum-Welch re-estimation procedure to completion on each of the K observations to estimate the starting node as above and take a simple average of the starting node distribution over all sequences. (Note that if the source model were known, it would be simple to use the Viterbi algorithm [2] to estimate the starting node.)

A well known problem with this method is that it only finds a locally optimal solution. Thus methods capable of broader analysis of the search space are sought.

2.2   Ensemble Averaging Methods

The second approach described here is a special case of the method suggested by Mackay [12] where an ensemble of models is trained. In this paper, one model is estimated for each of the K observation sequences (other approaches are possible but are not considered here). This enables the formation of K independent model estimates from the training sequences. From these, the next step is to examine the efficacy of combining the independent parameter estimates using a range of simple averaging techniques of the

following form:

$$\overline{a}_{ij} = \frac{\sum_k W_k a_{ij}^{(k)}}{\sum_k W_k} \tag{3}$$

$$\overline{b}_{ij} = \frac{\sum_k W_k b_{ik}^{(k)}}{\sum_k W_k} \tag{4}$$

$$\overline{\pi}_i = \frac{\sum_k W_k \pi_i^{(k)}}{\sum_k W_k} \tag{5}$$

where W_k is the weighting factor for each sequence and $\lambda^{(k)} = (A^{(k)}, B^{(k)}, \pi^{(k)})$. The quality of all model estimates is judged by the probability of that model generating an unseen set of test sequences from the same source as the K training sequences as described below.

The intention is to improve on Rabiner and Juang's method described above using a weighted combination of an ensemble of learnt models to avoid local minima traps as much as possible. Although each sequence is matched to the structure of each model, structure is not incorporated in the averaging step itself. It therefore places a limited reliance upon structure information.

This paper only focuses upon uniform weighting methods in the search for the best model. For an evaluation of other weighting schemes see [8].

2.3   Viterbi Path Method

The Viterbi-path training algorithm is as follows. For a vector of training sequences $o_i(t)$ for time $t \in [1, T]$ (where T is the sequence length and i is the sequence index) and a specified HMM model structure we perform the following training procedure:

Algorithm 3.1: (A_n, B_n, π_n)=Viterbi Path Counting($A, B, \pi, o, loops, num\_seq$)

Initialisation:

1. Create counter matrices A_c, B_c, π_c by setting their elements to be 0 if the corresponding element in A, B, π respectively is zero, and set them to a user-specified initial seed c_0 otherwise as defined by (6).

Recursion:

2. Iterate as follows:

    for $i \leftarrow 1$ **to** $loops$ **do**

        for $j \leftarrow 1$ **to** num_seq **do**

            2.1. Define temporary HMM matrices A', B', π' to be a function of the normalized counter matrices as defined by (7-9). These are used to locate the best path estimate for the current training sequence and incorporate additional randomness to help avoid local minima traps.

            2.2. Use the Viterbi algorithm with the temporary matrices A', B', π' to select the most likely path $w(t)$ for the current observation sequence o_i

            2.3. Add 1 to $\pi_c(w(1))$ to increment the counter for p at the first node $w(1)$ in the most likely path

            2.4 Add 1 to $B_c(w(1), o(j, 1))$

            2.5 Loop through the current observation sequence, adding 1 to the values in A_c for every transition from $w(t)$, and adding 1 to B_c for every symbol emission w.

Termination:

3. Return the final estimate A_n, B_n, π_n of the model parameters to be the normalized counter matrices, where the subscript n is a label denoting normalization.

    The parameter $loops$ is a fixed number of iterations chosen by the user. Other means of

determining the stopping point of the algorithm exist including measures quality of fit to the training data. However this method of determining the stopping point is simple and has produced reliable results on the trials to-date. The parameter num_seq is the number of training sequences in the training set.

The parameter

$$c_0 = initial\_val * loops \qquad (6)$$

may be adjusted to improve the performance. Increasing $initial\_val$ assigns a higher weight to the initial count values than to the new data.

The term $recurrent\_frac$ describes a factor used for a randomizing term that is added to the ongoing estimates A', B', π' of the model parameters in order to help avoid local maxima traps during the training process. This does not alter the counter matrices, but it does allow the selection of the best path some more flexibility. The usage is as follows:

$$A' = normalise\left(Ac + \frac{max(Ac)}{recurrent\_frac} * model\_structure. * rand(N, N)\right) \qquad (7)$$

$$B' = normalise(B_c) \qquad (8)$$

$$\pi' = normalise(\pi_c) \qquad (9)$$

where $model\_structure$ is a binary mask which determines whether the model is left-right, cyclic, or any other structure. This is determined at the start of the training process from the non-zero elements of the initial seed model. The term $rand(N, N)$ is an $N \times N$-dimensional matrix of random numbers in the interval $[0, 1]$ which is designed to reduce the likelihood of local maxima traps during convergence.

This method simplifies the training process because it simply counts states and transitions, and estimates the model parameters from those total counts. The hard work of finding the best path is hidden within the call to the Viterbi algorithm. In addition, the dependence upon the initial model is minimized because the model starts out in a maximum entropy state. It should be compared to the Best-First model merging method of Stolcke and Omohundro [14] but instead of merging models, it iteratively inserts the training sequences into a single fixed model structure using Viterbi's algorithm to select the path to be incremented.

In contrast, Rabiner and Juang's method relies on complex re-estimation formulae which make it difficult to find globally optimal solutions and there is a high degree of dependence upon the initial seed value.

There are three parameters which may be adjusted in this algorithm. The number of iterations is an important parameter. The initial counter values of the matrices may be adjusted by the $initial_val$ parameter. Finally, the choice of path may be selected using models other than the counter models — for example, a random model could be added to the counter model to slow down the rate of convergence and perhaps avoid local maxima traps (this third option is implemented in this paper using the $recurrent\_frac$ parameter, which contributes a specified multiple of a random model with the correct transition structure to the current estimate A_c before normalization).

Note that the method is not a standard EM-type training algorithm because the update rule depends upon local counts of the number of modifications of each branch of the model. In contrast, EM algorithms operate upon the probabilities from the previous iteration alone. Another difference from standard EM-type methods is that the sequences

are presented individually in turn, rather than simultaneously.

It is also interesting to note that the Viterbi path counting method depends upon the order of the training sequences, unlike the other methods investigated in this paper. The question of whether sequence ordering makes a significant difference to learning will be the subject of future research. The EM algorithm also has another property which the Viterbi Path Counting algorithm doesn't have; the joint probability of the training patterns is guaranteed to increase with each iteration.

2.4   HMM Diagnostics

McCane and Caelli [11] proposed a set of methods for ascertaining the characteristics of a given HMM based upon its parameters in terms of three key indicators: 1) The HMM Condition Number, 2) Residual Sum Vector, and 3) Conditional Information Content.

These measures provide an indication of the contributions of the different parts of the HMM to the properties of the model in terms of predicting observation and state sequences. The condition number is calculated from the concatenation of the A and B matrices, and indicates the likelihood of generating the correct state sequences using row independence on the augmented matrix of A and B as the measure.The residual sum matrix identifies which states or observation sequences need to be refined or removed from the model to improve it. The last measure is based on Mutual Information and identifies if the HMM is likely to perform better than a simple Bayesian classifier (trained with Maximum Likelihood techniques) using the B matrix alone.

This paper focuses upon the condition number and compares it with the standard measures of model quality based on the train and test methodology.

3 Train and Test Methodology

For each parameter set, seed models were generated (the Initial Generating Models, or IGMs), and used to generate ensembles of sequences. Ensemble learning and Rabiner and Juang's method were compared using the train-and-test paradigm [8]. The method of model quality evaluation for a model M using a test sequence set S was to evaluate the *model quality*, given by

$$Q(M) = \prod_{s \in S} p(s|M)$$

for sequence probabilities $p(s|M)$. In other words Q(M) represents the joint probability of all test sequences S being generated by model M.

The methods were compared over a range of parameters. The range was selected to correspond with situations that arise most commonly in the use of HMMs. Only left-right models were investigated.

The training set contains $trials$ samples, where a maximum of $mcfrac$ samples are really different, and the rest are duplications of the $mcfrac$ samples. The test set contains $unseen\_trial$ samples. The whole process of training and testing is repeated $n\_repetitions$ times for each model. There were 10 initial generating models created randomly for each parameter setting. This is a relatively small number but we believe that the number was sufficient given the consistency of results and overall large number of parameter sets studied.

The test process used exactly the same number of trials as the training process, so $trials = unseen\_trials$.

In all experiments, the number of loops for Viterbi Path Counting was $loops = 10$, the

13

initial value parameter for Viterbi Path Counting was $initial\_val = 5$ and the recurrent fraction was $recurrent\_frac = 1.25$.

Two classes of trial were conducted, involving the variable pairs $(N, mcfrac)$ and (M, seq_length) respectively, keeping all other variables fixed. Tables 2 and 3 show the parameters used in each trial.

3.1 Results and Discussion

The relative training method quality indicator is defined as the average quality of ensemble learning minus the quality of Rabiner's method (using log probabilities for unseen sequence sets):

$$P_d = \mathrm{logQ(Ensemble)} - \mathrm{logQ(Rabiner)}$$

The results were scaled to improve the comparison by dividing the log probability differences P_d by the number of sequences and by the sequence length to form a new normalised parameter P which describes the per-symbol difference in a way which treats all parameter combinations on an equal footing. The results are shown in figures 4, 5, and 6.

The analysis gives an indication of the boundary between regions in which ensemble learning is superior to Rabiner's method, and vice versa. These results spanned a broad range of parameters and are sufficiently consistent to suggest that a general trend has been found with this technique.

In general, for arbitrary left-right models, it is best to use Rabiner's method when the number of sequences is less than or equal to 2, and use ensemble learning for 3 or more sequences. However the boundary is quite irregular, and depends also upon all
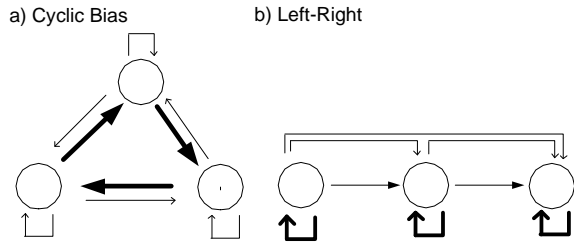
14

Figure 1: Cyclic and Left-Right structures. Bold arrows indicate higher probabilities on average. An absence of an arrow connecting a vertex pair indicates a forbidden (zero-probability) transition.

Quantity	Value
Number of distinct training sequences (before duplication) : mcfrac	$(1, \ldots 8)$
Number of nodes: N	$(3, \ldots, 12)$
Number of observation symbols: M	6
Sequence Length: $\mathrm{Seq\_length}$	5
Number of trials used in the training process: Trials	12
Number of trials used in the testing process: $\mathrm{Unseen\_trials}$	Trials

Figure 2: The parameter set for the $(N, mcfrac)$ trial

Quantity	Value
Number of distinct training sequences (before duplication) : mcfrac	4
Number of nodes: N	5
Number of observation symbols: M	$(3, \ldots, 12)$
Sequence Length: $\mathrm{Seq\_length}$	$(4, \ldots, 15)$
Number of trials used in the training process: Trials	12
Number of trials used in the testing process: $\mathrm{Unseen\_trials}$	Trials

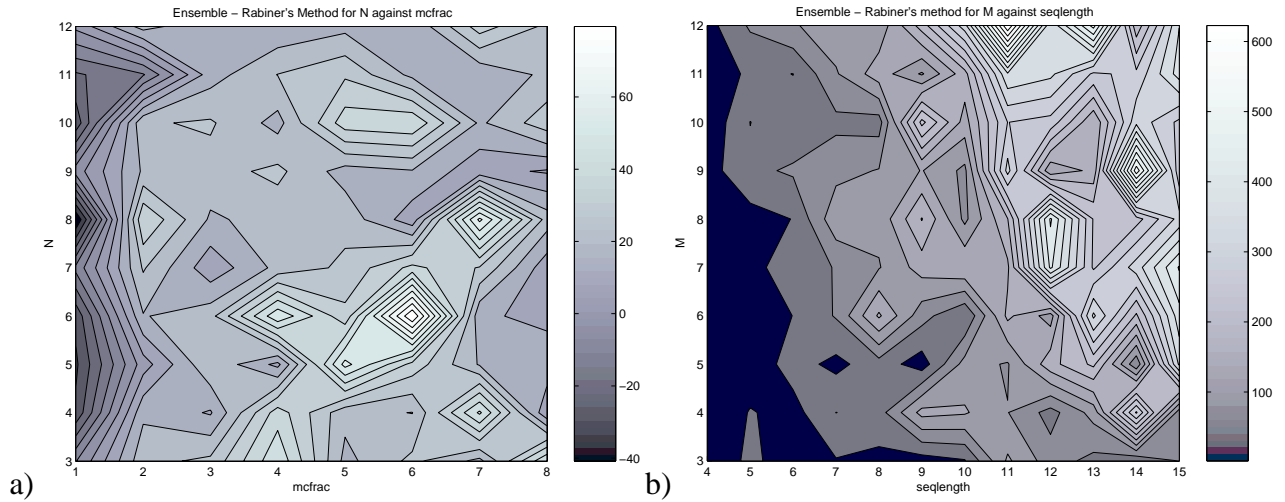Figure 3: The parameter set for the $(M, seqlength)$ trial

a)   b)

Figure 4: Relative merits of Ensemble Averaging against Rabiner's (EM) method.
(a) Relative training method quality P is shown as a function of model size N and
$mcfrac$, holding all other parameters fixed. For $mcfrac = 1$ the best method is Rabiner's method, but for $mcfrac > 1$ the best method is ensemble averaging. (b) Relative
training method quality P is shown as a function of number of symbols M and training
sequence length $seqlength$, holding all other parameters fixed. Ensemble averaging is
always marginally better than Rabiner's method in this figure, and for larger values of
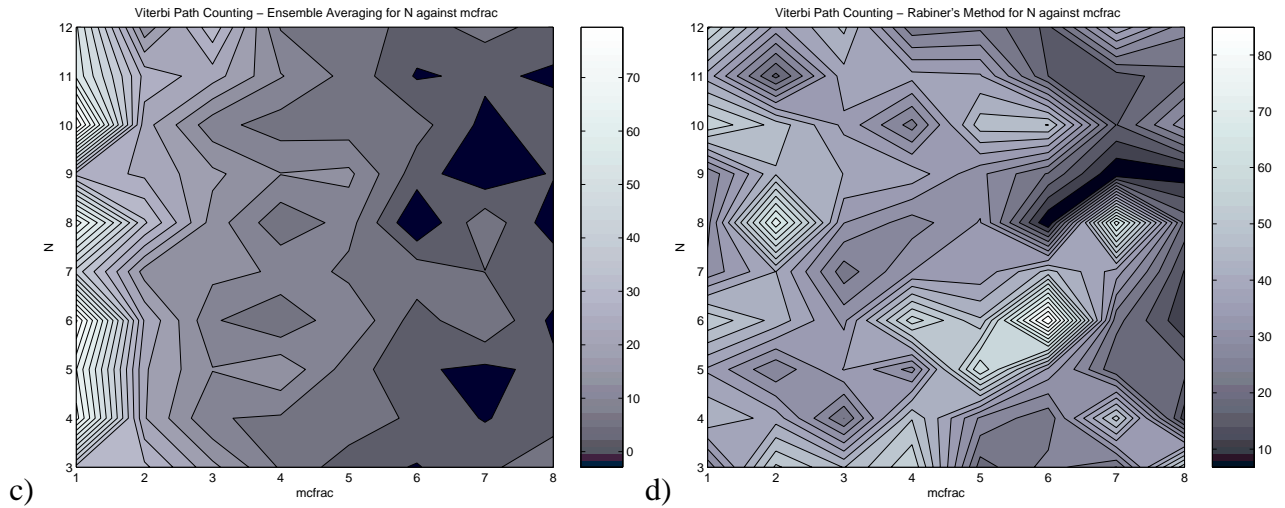these parameters this advantage is further increased.

c)   d)

Figure 5: Comparing Viterbi Path Counting with other methods using N and $mcfrac$.
(c)Viterbi Path Counting - Ensemble Averaging against N and $mcfrac$. The weakness
of the ensemble averaging method on single sequences is apparent. The Viterbi Path
Counting method does not suffer from these weaknesses. (d) Viterbi Path Counting -
Rabiner's method against N and $mcfrac$. Superior learning performance of the Viterbi
Path Counting method is visible over all parameters in the figure, although the degree of
superiority varied substantially.

parameters, so for between 2 and 5 sequences it seems wise to run preliminary trials to determine which method will be best with the particular model structure in question as the best method seems to depend upon the model size.

The figure provided could be used to select the best method, or else the methods could be compared on test cases as was done in this paper and the most appropriate method for the particular model geometry could be used for specific situations. This gives a simple way of discriminating between the alternative training methods, based on the train-and-test paradigm. As a general rule, if there are two or more training sequences then ensemble averaging will outperform Rabiner's method for the models investigated in this paper.

4   HMM Condition Number Methodology

McCane and Caelli [11] have investigated methods for determining the type of an HMM using three indicators of redundancy in the A and B matrix structure. The Residual Sum Vector and Conditional Information Content measures were beyond the scope of this paper. However the HMM Condition Number measure was evaluated on the initial generating models and on the corresponding trained models. From these calculations, two issues were checked:

1. That both the IGMs and the trained models satisfy the criteria for being a well-conditioned HMM.

2. That the quality and condition number indicators correlate with each other.

From these observations we would expect to conclude that respectively either

1. Our trial is appropriate for the evaluation of HMM learning methods (from the IGM indicator functions), or

2. Our trial is inappropriate for the evaluation of HMM learning methods (from the IGM indicator functions).

Additionally, either

1. The methods train well (with some more confidence from the correlations in the indicator functions between the IGM and the trained models), or

2. The methods don't train well (as measured by the correlation between the IGM and the trained model condition number values) even if the sequence families do appear to overlap.

In practice, the inverse condition number (ICN) is used for analysis because it is lies in the convenient range [0,1]. In [11], it has been conjectured that the best performing HMMs should be those for which the rows of the A and B matrices are nearly linearly independent (i.e., large values of ICN) as this enables maximum discrimination between the state and observation symbols. To calculate the ICN, we form the row augmented matrix $C = A|B$ and perform singular value decomposition on C. The more independent the rows of C, the less ambiguous the HMM is. We therefore define the inverse condition number by

$$\gamma^{-1} = \frac{\sigma_{min}}{\sigma_{max}}$$

where $\sigma_{min}, \sigma_{max}$ are the minimum and maximum singular values respectively. A value of 1 indicates a high degree of independence, and a value of 0 indicates a low degree of

independence. It is important to note that because only the largest and smallest singular values are included, there is potential for a significant portion of A and B to have no effect upon the condition number.

Table 7 shows the parameters used in the trial. The following quantities were evaluated for these parameters with Rabiner's method, the Ensemble Learning method, and the Viterbi Path Counting method:

- Average inverse condition number of initial generating models.

- Average inverse condition number of trained models.

- Average model quality on unseen samples.

4.1 Results and Discussion

The trials were repeated 10 times for left-right models and the results are shown in figure 8.

We found that the ensemble learning method and the Viterbi Path counting produced results with lower inverse condition number than the Rabiner method. Despite this, better quality models (as measured by the fit to the unseen sequence set) were produced as was reported in previous papers [8]. Although the inverse condition number is a guide to how predictive or specific the model is (see [11]), from our trial we can conclude that it does not give an indication of the quality of fit to unseen data.

Figure 9 shows the relationship between IGM and the trained models, measured using the inverse condition number on left-right models. Figure 9 provides some insight into the behaviour of learning in HMMs. Despite the disparity in quality, the condition numbers
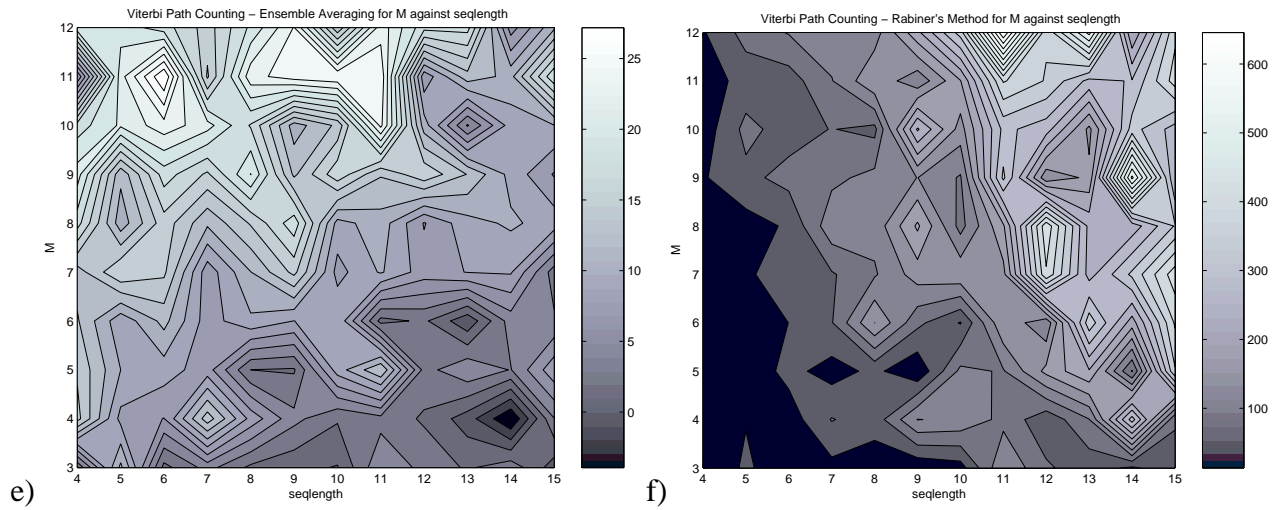
Figure 6: Comparing Viterbi Path Counting with other methods using M and $seqlength$. (e) Viterbi Path Counting - Ensemble Averaging against M and $seqlength$. The Viterbi Path Counting method is superior over almost all parameters tested here, but displays the greatest advantage over Ensemble Averaging in the case of short sequences and large numbers of observation symbols. (f) Viterbi Path Counting - Rabiner's method against M and $seqlength$. The Viterbi Path Counting method is superior over all parameters shown, and displays the greatest advantage over Rabiner's method for longer sequences and larger numbers of symbols in this experiment.

Quantity	Value
Number of states N	5
Number of symbols M	6
Sequence length $seqlength$	5
Number of sequences in the training set $Trials$	6
Number of distinct training sequences $mcfrac$	6

Figure 7: Parameters for the Condition Number Trial

Model/Method	Av. LR Quality	Av. LR ICN
Initial Generating Model	-28.8	0.71
Rabiner's method	-59.2	0.062
Ensemble learning	-41.8	0.12
Viterbi Path Counting	-34.6	0.090

Figure 8: Table of results for Quality and Inverse Condition Number study

showed similar values. This means that the condition number is not necessarily a good guide to the performance of a model.

Note that the superior Viterbi path counting method produced a more even range of condition numbers. The results suggest that most HMM training from observation sequences has so little structural information to work from that there is no way the model can be correctly deduced from the sequence set. Figure 9 illustrates this point — even for the best method with high quality of prediction the amount of information in unused nodes tends to make the condition number measure a poor indicator of model quality. The figure does however show that path counting has a very even condition number spread, which makes good sense from this perspective as this method concentrates upon improving what is known about a very small set of critical nodes at the expense of the bulk of the other nodes which play almost no part in defining the characteristics of the model.

These results display a different characteristic pattern. Whilst all inverse condition numbers are very low, ensemble averaging now produces better-conditioned models than Viterbi Path Counting, which retains the same inverse condition number values. This is despite the fact that Viterbi Path Counting produces better quality models. This confirms the hypothesis that large parts of the model play little part in determining the performance of the model in classification, yet they still affect the condition number.

This in turn suggests that a measure of the quality of conditioning of the HMM should be defined to remove this problem of high sensitivity to nodes which don't play a large part in the model's performance. This should take into account the model structure so that low-probability paths are ignored.

From these results we observe that Viterbi Path Counting does not produce a wide

range of condition numbers compared with the other two methods.

5 Model Quality Study

For each parameter set, seed models were generated (the Initial Generating Models, or IGMs), and used to generate ensembles of sequences. Ensemble learning, Viterbi path counting and Rabiner's multi-sequence (single-convergence) method were compared using the train-and-test method [8]. The method of model quality evaluation for a model M using a test sequence set S was to evaluate the *model quality*, given by

$$Q(M) = \log\Pi_{s \in S} P(s|M)$$

The methods were compared over two sets of parameters. The parameters were selected to correspond with situations that arise most commonly in the use of Hidden Markov Models. Only left-right models were investigated. Figure 10 shows the combinations of parameters tested. The main variable of interest was the number of distinct training sequences, which was set to 1 and 5.

The test process used exactly the same number of trials as the training process. Averages were formed using 5 models for each parameter set. Seed models were generated randomly. Test and training sequences were then generated from the seed models.

5.1 Results and Discussion

We see that when training from 5 distinct observation sequences (no duplication), the Viterbi-Path method performs as well as the ensemble learning method, and both outper-
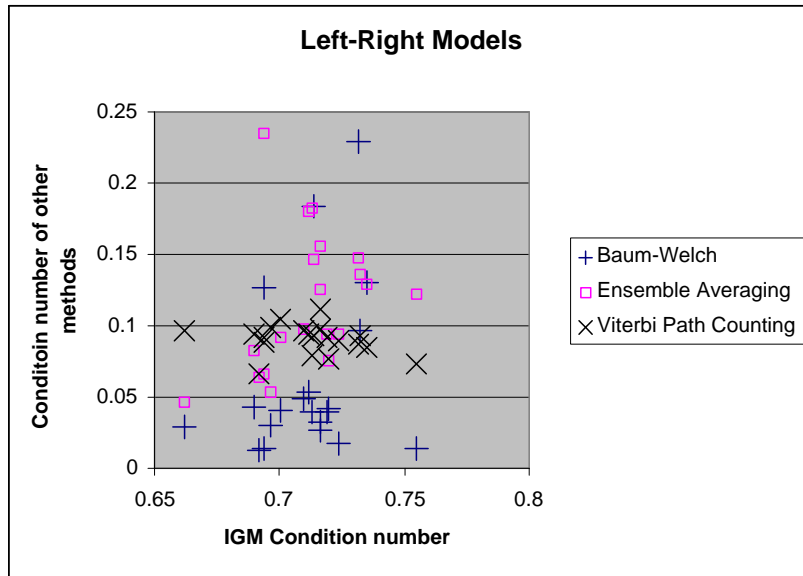
Figure 9: The relationship between the inverse condition numbers of models produced by the Baum-Welch, Ensemble averaging, and Viterbi Path Counting methods, plotted against the inverse condition number of the initial generating models.

Quantity	Value
Number of distinct training sequences (no duplication) : $\mathrm{Model\_frac}$	$1, 5$
Number of nodes: N	6
Number of observation symbols: M	4
Sequence Length: $\mathrm{Seq\_length}$	5
Number of sequences used in the training process: Trials	10
Number of sequences used in the testing process: $\mathrm{Unseen\_trials}$	10
Number of trial repetitions for averaging: $n\_repetitions$	50

Figure 10: Parameters for the Model Quality Study

Method	Average Q	Std.Dev. of Q
True	-35.1	4.0
Viterbi Path	-48.5	7.4
Ensemble Averaging	-54.7	9.1
Rabiner Merge	-91.4	35.4

Figure 11: Training on 5 different sequences.

form Rabiner's method (see table 11). For multiple copies of a single training sequence, an even better result for Viterbi Path was obtained, shown in table 12. The Viterbi Path method also produced models with better quality standard deviation. Similar results were obtained using larger numbers of sequences, both distinct and duplicates.

Figure 14 shows the relative performance and execution times using the classification performance (measured in terms of the number of correct models) using Left-Right models. The parameters used in this trial are shown in figure 13. It also shows the algorithm execution times for each method.

In an attempt to confirm the simulated data performance results on real-data, we developed a video gesture recognition system which could recognize 26 gestures corresponding to the letters of the alphabet. There were 20 training samples and 10 test samples for each distinct gesture; so there were 780 gesture videos in the database in total. After trajectory estimation and smoothing, the angle of movement of the center of the hand was broken into discretized into 1 of 18 directions over the 25 frames of video to form the discrete observation sequences.

The performance of HMMs obtained via Baum-Welch, Ensemble Averaging, and Viterbi Path training methods were compared using this video dataset. Surprisingly all three methods yielded quite high (80-90%) average recognition rates — there were no significant performance differences between the three methods.

This suggests that the real data being generated is being distinguished primarily using B-matrix information (i.e., observation statistics), and thus it is not a particularly effective test of A-matrix learning ability (i.e., a-priori knowledge of structure).

This is consistent with the design of the three algorithms, since they each have different

24

Method	Average Q	Std.Dev. of Q
True	-27.6	2.1
Viterbi Path	-64.8	14.6
Rabiner Merge	-77.5	19.1
Ensemble Averaging	-84.0	23.9

Figure 12: Training on 5 copies of the same sequence.

Parameter	Value
Number of sequences used in classification	1000
Classification method used	Two-class classification
Parameter values and test set size	See figure 10.

Figure 13: Parameters and data size used in the classification-based study of algorithm execution times.

Item	True Model	Baum-Welch	Ensemble	Viterbi Path
VPC $initial\_val = 15$	652.9	547.3	633.4	627.9
Mean time to converge (sec)		3.9	0.25	0.23
VPC $initial\_val = 5$	670.8	553.1	644.7	669.8
Mean time to converge (sec)		4.3	0.27	0.22

Figure 14: A range of relative training method classification scores are shown, together with training times for comparison. The $initial\_val$ count seed in Viterbi Path Counting doesn't significantly affect the convergence time, which was faster than all other methods tested. Note that setting the initial count values to high levels limits the learning quality of VPC given the fixed-iteration termination criterion. These times were obtained using MATLAB code on a 1.5Ghz Pentium 4 with 512Mb of RAM.

approaches to learning structure, and similar approaches to symbol statistics estimation. To investigate this hypothesis the authors intend to explore a diverse range of real datasets in an attempt to construct more demanding tests of structure-learning ability.

6 Conclusions

The observation that Ensemble averaging lies between Viterbi Path and Rabiner's method reflects the fact that the method takes partial account of structure in learning (as each model is trained using Baum-Welch but the averaging step does not incorporate structure). There appears to be a complicated boundary defining the region in which Rabiner's method is superior to the ensemble learning method for small numbers of observation sequences.

The surprisingly good performance of the Ensemble averaging method (which ignores node alignment) can be attributed to the high level of randomness in the structure of the initial generating models. For more than 3 observation sequences, the Viterbi Path counting learning method is generally the best method, followed by Ensemble averaging and Viterbi learning, and for fewer than 3 sequences, Rabiner's method is generally best. Viterbi Path Counting outperforms ensemble averaging in nearly all situations.

These results demonstrate that the size and structure of an HMM, and the training set size and sequence length determine the HMM's training characteristics and they also determine the choice of the best training method in a fairly complex way which can only be ascertained by comparative studies such as the one presented in this paper, or preferably analytic results where available.

The path counting algorithm was the best method on nearly all parameters tested (with

the exception of single-sequence datasets in which case Rabiner's method was best). This suggests that the principles incorporated into its design are important. These will be studied in more detail in future work, which may include a study of the algorithms on continuous HMMs as well as a study of their effectiveness on real applications such as speech or handwriting recognition. The Viterbi Path method has recently been compared with Baum-Welch on a hand gesture recognition system and produced superior results which will be reported in future work.

For initial generating models (IGMs) with higher inverse condition numbers, then all three methods produce very low inverse condition numbers. The inverse condition number does not provide a guide to model quality — a poorly conditioned model may fit the training data better than a well-conditioned model if both are trained on the same set of training data.

We observed that the Viterbi path counting algorithm produces a small range of inverse condition numbers. In contrast, Baum-Welch and Ensemble Averaging produce a wider range of condition numbers, suggesting that they are less efficient at learning. The Viterbi Path Counting algorithm produced models of the highest quality and classification accuracy in the shortest time, which also suggests that it makes the most efficient use of the data.

A test on a gesture recognition system currently under development was conducted. Although we had expected methods such as VPC to perform significantly better than Baum-Welch estimation, it turned out that similar levels of high recognition accuracy were obtained for all three estimation methods. This may be because this particular recognition task is relatively easy, so all methods do quite well. This result is however consistent

with the idea that averaging takes 'partial' account of the structure information. Since there appears to not be much structure information in the gesture data, additional tests on a wider range of methods would help confirm this. It was encouraging that the newer methods performed no worse than the commonly used Baum-Welch method.

Further refinements to the algorithm itself, such as the use of a range of alternative path-selection techniques, the introduction of variable rates of learning, and adapting the implementation according to the model structure and training data set are currently being investigated.

References

[1] L. R. Rabiner, *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*, Proceedings of the Institute of Electrical and Electronics Engineers (IEEE), pp. 257-286, Volume 77, Number 2, February 1989

[2] L. R. Rabiner, and B. H. Juang, *Fundamentals of Speech Recognition* New Jersey Prentice Hall, 1993.

[3] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi, *An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition*, The Bell System Technical Journal pp. 1035-1074, Volume 62, Number 4, April 1983.

[4] J. Rottland, C. Neukirchen, D. Willett and G. Rigoll, *Large Vocabulary Speech Recognition with Context Dependent MMI-Connectionist/HMM Systems Using the WSJ Database*, Proceedings of the 5th European Conference on Speech Com-

munication and Technology, Rhodes, Greece, pp. 79–82, 1997, available at cite-seer.nj.nec.com/rottland97large.html.

[5] Andrew D. Wilson, Aaron F. Bobick, *Hidden Markov Models for Modeling and Recognizing Gesture Under Variation*. International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI), Volume 15, Number 1, February 2001, pp. 123-160

[6] A. Nefian and M. Hayes, *Hidden markov models for face recognition*, Proceedings of the International Conference On Acoustics, Speech and Signal Processing (ICASSP98), pp. 2721–2724, Seattle, United States of America, 1998. Available at citeseer.nj.nec.com/236258.html.

[7] J. J. Lee, J. Kim, and J. H. Kim, *Data-Driven Design of HMM Topology for Online Handwriting Recognition*, International Journal of Pattern Recognition and Artificial Intelligence, Volume 15, Number 1 (2001) pp. 107-121. World Scientific Publishing Company.

[8] R. I. A. Davis, and B. C. Lovell, and T. Caelli (2002) *Improved Estimation of Hidden Markov Model Parameters from Multiple Observation Sequences*. Proceedings of the International Conference on Pattern Recognition (ICPR2002), August 11-14 II, pp. 168-171, Quebec City, Canada.

[9] R. I. A. Davis, C. J. Walder, and B. C. Lovell. (2002) *Improved Classification Using Hidden Markov Averaging From Multiple Observation Sequences*. In the Proceedings of the Workshop on Signal Processing and Applications (WOSPA2002) Brisbane, Australia, 2002.

[10] R. I. A. Davis and B. C. Lovell, *Improved Ensemble Training for Hidden Markov Models using Random Relative Node Permutations*, Proceedings of the Workshop on Digital Image Computing (WDIC2003), pp. 83-86, February 7, Brisbane, Australia, 2003

[11] T. Caelli, and B. McCane, *Components analysis of hidden Markov models in Computer Vision*, 12th International Conference on Image Analysis and Processing, September 17-19, 2003 Mantova, Italy (In Press).

[12] D. J. C. Mackay, *Ensemble Learning for Hidden Markov Models*, *Technical report*, Cavendish Laboratory, University of Cambridge, UK, 1997.

[13] X. Li, M. Parizeau, R. Plamondon, *Training Hidden Markov Models with Multiple Observations — A Combinatorial Method. IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 22, No. 4, pp. 371-377, April 2000.

[14] A. Stolcke and S. Omohundro. *Hidden Markov Model induction by Bayesian model merging*, Advances in Neural Information Processing Systems, pp. 11-18, Morgan Kaufmann, San Mateo, United States of America, CA1993.

[15] G. Elidan, M. Ninio, N. Friedman, and D. Schuurmans *Data perturbation for escaping local maxima in learning.* Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-02), 2002.

[16] Z. Ghahramani, *An Introduction to Hidden Markov Models and Bayesian Networks*, International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI), pp. 9-42, Volume 15, No. 1, February 2001.

[17] T. Caelli, A. McCabe, G. Briscoe, *Shape Tracking and Production Using Hidden Markov Models.* International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI), pp. 197-221, Volume 15, Number 1, February 2001.

[18] S. Müller, S. Eickeler, G. Rigoll, *An Integrated Approach to Shape and Color-Based Image Retrieval of Rotated Objects Using Hidden Markov Models*, International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI), pp. 223-237, Volume 15, Number 1, February 2001.

[19] C. Lee, Y. Xu, *Online, Interactive Learning of Gestures for Human/Robot Interfaces*, pp. 2982-2987, IEEE International Conference on Robotics and Automation, Minneapolis, United States of America, Volume 4, 1996.

[20] C. J. Walder, R.I.A. Davis, *IRIS source for estimating Hidden Markov Models*. C code available at: http://www.itee.uq.edu.au/ iris/CVsource/source.html

[21] K. P. Murphy et al. *The Bayes Net Toolbox for Matlab* This version was last updated on 28 July 2002. Available at http://www.cs.berkeley.edu/ murphyk/Bayes/bnt.html.